片上多核处理器验证综述

沈海华

摘要：片上多核处理器是当今主流处理器结构，也是最复杂最难验证的一类集成电路芯片。本文梳理了集成电路验证领域现有的技术和工具，结合国内外片上多核处理器验证的实例，分析了现有验证技术和工具面对片上多核处理器验证的适用性和局限性，并针对现存问题进行了初步探讨。

关键词：集成电路，片上多核处理器，验证，仿真

1 引言

1.1 验证危机

纵观集成电路工艺发展历史，自上世纪 90 年代中的 0.25 微米开始，历经 0.18 微米（1999 年）、0.13 微米（2001 年）、90 纳米（2003 年）、65 纳米、45 纳米、28 纳米、22 纳米工艺 (2009 年，英特尔)，半导体工业突破了一个又一个看似不可能跨越的瓶颈，神奇地继续遵循着摩尔定律高速发展。半导体技术的发展使电子产品设计越来越复杂，相应地设计中出现错误的风险也越来越高。为了保证设计的正确性，理想情况是通过各种验证技术手段遍历待验证设计（Design under Verification, DUV）的全部状态空间。然而，当前集成电路规模普遍在千万门以上，包含上亿晶体管的设计已不再罕见，遍历全部待验证设计状态空间已成为“不可能完成的任务”。如何用有限的技术手段，在有限的时间内验证几乎膨胀到“无限”的待验证设计状态空间成了摆在所有集成电路从业者面前最困难的问题。十几年来，验证领域科研人员和工程师开展了大量的研究，在模拟仿真、形式化方法、仿真加速、状态空间裁减、状态空间覆盖度量等方面取得了一系列研究成果。尽管如此，面对复杂集成电路设计带来的越来越严重的“状态空间爆炸”，现有技术手段仍显得捉襟见肘。早在 2003 年，国际半导体技术路线图的作者就深有感触地称验证已经“由瓶颈转化成了危机”。到了今天，集成电路芯片首次流片成功率（此处指首次流片错误率达到可用水平）已越来越低，“验证危机”成为业内共同担忧的问题。面对现实，即使英特尔、IBM、AMD、英伟达（NVIDIA）等国际知名芯片厂商也不得不正视首次流片错误率问题，而硅后验证更成为近两年验证领域研究的热点。

纵观集成电路验证技术的发展历史，如果说在上世纪 90 年代验证还只是各种设计工程师的“兼职”工作，那么在 20 年后的今天，验证已经成为集成电路设计流程中不折不扣的“瓶颈”和“危机”，是集成电路厂商、电子设计自动化（EDA）工具厂商及学术界高度重视的重点研究领域。本文重点关注当今最复杂的一类集成电路—片上多核处理器（Chip Multi-Processor，CMP）的验证。

1.2 片上多核处理器的发展及其对“验证危机”的影响

片上多核处理器是在单个芯片上设置多个处理器核构成的多处理器系统，应用程序的多个线程可以在多个处理器核上同时执行[1, 2]。在过去的 10 年中，片上多核处理器实现了从体系结构实验室的宠儿到商业化处理器的跨越。今天，处理器设计已全面步入片上多核处理器时代。

IBM、英特尔、AMD 和 ARM 等国外主流微处理器厂商纷纷支持单片多核处理器结构。

IBM 公司的微处理器主要为 Power 系列处理器[3~6]，用于高性能服务器和高性能计算机。IBM 于 2001 年发布了双核的 Power4，片内集成两个超标量的类 Power3 处理器核。2004 年 IBM 发布了 Power5 双核同时多线程微处理器。2007 年 IBM 发布了双核的 Power6，采用 65nm 绝缘体上外延硅（Silicon-on-insulator，SOI）工艺，主频高达 4~6GHz。2010 年推出 Power7 八核 32 线程处理器，采用 45nm 工艺，每个核支持四个线程，每个处理器核的性能达到 32GFlops。Power7 用于美国国防部高等研究计划局（DARPA）的万亿次（Petaflops，10^{15}flops）超级计算机。据称 IBM 后续的芯片项目 Power 8 处理器的设计已经启动，将具有更高的性能和集成更多的处理器核。

英特尔公司的微处理器主要包括安腾和 x86 系列。其中安腾系列用于服务器，X86 系列用于桌面和服务器。英特尔在 2008 年发布了四核的安腾处理器 Tukwila，2011 年在 ISSCC 上公布了八核的安腾处理器 Poulson，采用 32nm 工艺，片上集成了 31 亿个晶体管，片上高速缓存高达 54MB；在 X86 系列英特尔于 2006 年推出了基于 Core 构架的处理器 Conroe；2008 年推出基于 Nehalem 结构的四核处理器，2009 年公布了基于 Nehalem 结构的八核至强处理器，2009 年底推出了把 CPU 和 GPU 封装在一起的多核处理器（Core i7），其中 CPU 采用 32nm 工艺，2011 年推出基于 Sandy bridge 构架的新一代多核处理器，支持 256 位向量指令 AVX，处理器核的个数达到 8-16 个，并且将要采用 22nm 的工艺。此外，2010 年，英特尔还推出了众核实验芯片 SCC 单芯片云计算机，包含 48 个处理器核，具有较好的可伸缩性，未来可扩展到 1000 个处理器核。针对低成本电脑和移动计算领域，2008 年英特尔推出了 Atom 微处理器。

AMD 公司于 2007 年推出基于 K10 结构的四核处理器 Barcelona，2009 年下半年推出代号为 Istanbul 的六核心 45 纳米 Opteron 处理器，2011 年推出基于 Bulldozer 架构的 12-16 核的 Interlagos 处理器、以及集成处理器核和图形显示核心的融合加速处理器 Fusion。在移动计算领域和低成本电脑领域，AMD 也推出了 Bobcat 处理器。

ARM 公司的主要产品是 IP 核，通过设计出高效的 IP 内核，授权给半导体公司使用，产品包括 ARM7、ARM9、ARM10、ARM11 和 Cortex 系列等。ARM 系列产品占据了 32 位嵌入式处理器 75%的市场，在手机等移动终端领域更是处于垄断地位。2009 年 ARM 推出 Cortex A8 以及多核的 Cortex A9，采用超标量和乱序执行技术，主频可以达到 1-2GHz，开始主攻上网本和高端消费类电子市场。

在国内，通过“十五”和“十一五”期间的发展，在国家的大力支持下，我国已在微处理器设计方面打下了一定的基础，在微处理器研制方面取得了群体性突破，研制出多款通用微处理器和嵌入式微处理器。江南计算所、国防科技大学、中科院计算所、清华大学、北京大学等单位都取得了不少具有自主知识产权的成果。幸运的是，片上多核处理器结构的兴起进一步为国产处理器设计带来了新的机遇：与同时多线程结构相比，片上多核处理器结构内部的处理器核可以很简单，使得设计难度降低，在可用性、低功耗等方面也具有明显的优势，国内物理设计水平不高也仍可以实现。目前国内已经有多家单位从事片上多核处理器结构的研究。中科院计算所已先后成功流片了四核龙芯 3A 和八核龙芯 3B；江南计算所和国防科学技术大学等单位也研制出神威系列和银河飞腾系列处理器，在片上多核处理器研制上取得了优异的成果；清华大学在容错的片上多核处理器结构方面也开展了很好的研究；除上述单位外，还有许多单位围绕片上多核处理器的各项关键技术展开研究，限于篇幅，此处不再一一列举。

片上多核处理器结构的发展和流行很大程度上源于半导体工艺发展的推动。根据 2009 年国际半导体技术发展路线图（ITRS）预测，未来 10 年集成电路仍将按摩尔定律持续高速发展，2012 年将采用 22 纳米的工艺，2015 年将采用 17 纳米的工艺，2017 年将采用 14 纳米的工艺，2020 年将采用 10 纳米的工艺。ITRS 预测，到 2013 年，高性能微处理器芯片上可集成的晶体管将超过 88 亿个（2020 年超过 353 亿个），片上局部时钟频率将达到 7.3GHz（到 2020 年这一数字将达到 16GHz）。随着工艺条件的发展，片上多核处理器结构所包含的片上处理器核数目从目前的 2、4、8、16 核会一路增加到 64 核甚至更多，从多核(MultiCore)处理器发展为众核(ManyCore)处理器。多核的结构也从同构发展到异构，设计复杂程度达到前所未有的高度，随着片上多核处理器结构的日趋复杂，给处理器验证带来了巨大挑战，形成了日益严重的“验证危机”，表现为：

■ 状态空间更加复杂

设计规模和设计复杂度的增加使得片上多核处理器“状态空间爆炸问题”更加严重，与通用处理器相比，复杂的共享存储结构导致的不确定性给片上多核处理器验证雪上加霜。

■ 验证周期急剧增长

片上多核处理器巨大的设计规模向传统的验证方法和工具提出了巨大挑战，模拟仿真工具面临速度急剧下降、验证周期极大增长，形式验证工具则因“状态空间爆炸”加剧而变得不适用。共享存储结构验证和片上网络结构验证等新问题使传统验证方法和工具与不断增大的设计规模之间的矛盾愈加尖锐。

■ 调试及验证度量更加困难

验证度量是把握验证质量、管理验证流程的关键，覆盖率是传统验证度量标准之一。在片上多核处理器验证中，由于仿真加速设备的大量引入，以及首次流片错误率增加使得硅后验证蓬勃发展，在仿真加速验证和硅后验证环境下的覆盖率度量已成为目前验证领域亟待解决的问题之一。同时，硅后验证环境不像在软件模拟仿真环境那样，待验证设计内部状态具有可观测性，因此调试也成为了困扰验证人员的难题。

■ 验证成本极大增加

面对片上多核处理器的巨大设计规模，为了避免模拟仿真工具速度急剧下降导致的验证周期急剧增长，验证过程引入了昂贵的仿真加速工具，导致验证成本增加。而且，为了解决硅后验证的调试问题，引入了逻辑分析仪、甚至一些更昂贵的电学调试设备，如：红外线发射显微镜（IREM）、激光辅助调试、聚焦离子束（FIB）编辑器等，也极大增加了验证成本。此外，由于验证复杂度的增加，验证技术人员的投入成十倍的增加，人员开销及管理成本也成为验证成本急剧增长的重要来源。

以上只简单列举了片上多核处理器验证面临的挑战和问题，本文后续将详细分析验证领域现有技术及其国内外研究现状，并针对现有技术对片上多核处理器验证的适用性和局限性进行评述。

2 验证领域国内外研究现状及现有技术应对片上多核处理器验证的适用性和局限性

传统验证方法是否适用于片上多核处理器的验证？现有验证技术和工具是否足够应对

片上多核处理器的验证？要回答这些问题必须深入分析验证领域国内外研究现状，对现有的验证方法进行一一检阅。

2.1 硅前验证

传统的硅前验证方法包括模拟仿真方法、形式化方法和硬件仿真加速验证方法。

表1. 不同层次验证方法大致速度对比情况[8]

验证方法	仿真/运行频率(Hz)
系统模拟器	$\sim 10^3$
RTL[1]级模拟仿真	10^1 to 10^3
门级网表模拟仿真	10^{-1} to 10^1
硬件加速仿真	$\sim 10^5$
FPGA[2]原型仿真	$\sim 10^6$
硅后验证	10^7 to 10^9

模拟仿真方法是当前集成电路硅前验证应用最多，最常用的方法。与其它类型的设计相比，片上多核处理器在模块级模拟仿真应用上差别不大，只是相比普通超标量处理器模块规模更大一些（单模块最大规模可以是一个通用处理器核）。模拟仿真方法在片上多核处理器验证中的主要应用问题出在系统级模拟仿真上。虽然模拟仿真方法可以处理较大规模的设计，但速度并不理想。表 1 中列出了 ARM 的验证工程师统计的不同层次的软件模拟仿真、硬件仿真加速及硅后验证大致速度对比情况。可以看出，模拟仿真与硬件加速仿真/现场可编程门阵列/硅后验证相比，速度慢了多个数量级。面对设计规模不大的情况（如普通的单核处理器设计），可以通过投入更多的运算仿真资源，采用超大规模服务器农场来缓解模拟仿真方法的速度问题。例如英特尔为验证奔 4 用了 6000 台机器，24 小时不间断地运行了两年，所运行的总指令数也不过相当真正的奔 4 芯片运行两分钟的数量[7]。然而，对于片上多核处理器来说，随着设计规模和复杂度的增长，模拟仿真速度会进一步下降。在现有技术工具条件下，当待验证设计的规模和复杂度大到一定的程度，模拟仿真速度将会变得难以容忍。

目前常用的解决方法有两种：一是提高仿真器速度，这是许多电子设计自动化工具供应商不懈努力的目标，Synopsys 就多次宣称要保证其旗下的仿真软件 VCS 的模拟速度每两年都要有两倍至三倍的提升；二是增加仿真验证平台的规模和并行度，即采用更多的工作站和服务器，提高仿真过程的并行度，从而使得单位时间可以仿真的节拍（cycles）数增加。2008 年以来，面对传统模拟验证由于速度问题无法处理超大规模集成电路设计的尴尬局面，已有多家电子设计自动化工具供应商进行了提高并行度的各种方法的尝试。遗憾的是，到目前为止，投入的资源与获得的加速比相比，效果并不相称。虽有厂商宣称可提高模拟速度至单线程产品的 4~5 倍，从本文作者自身体验来看，对于真正的大规模设计，基本加速比为 1.5 左右，理论极限加速比难以达到。在电子设计自动化工具支持有限的情况下验证技术人员希望找出一种可行的方法来解决该问题，以便支持片上多核处理器等超大规模集成电路设计的模拟仿真，特别是系统级模拟仿真。在龙芯验证实践中，我们采用了层次化的模拟仿真验证方法来解决该问题，即把待验证设计逐级划分为更小的单元甚至模块，为各模块建立简单的参考模型，通过待验证设计与参考模型的自由变化、交叉、组合，既可以有效提高系统级模拟仿真的速度，又可以不断变化验证重点部位，最终实现全系统的模拟仿真。

覆盖率是评估仿真验证效果的重要指标，通过覆盖率分析可以指示验证的空白点，并指导此后的验证。仿真验证的过程从某种程度上来说就是不断运行更多的测试向量，追求更高的覆盖率，直到覆盖率达到 100%的过程。覆盖率通常包括代码覆盖率和功能覆盖率。代码覆盖率的确定依赖设计代码模型，在传统模拟仿真方法中，代码覆盖率通过模拟仿真工具就可以采集；功能覆盖率则需要验证设计人员自己定义模型和书写代码，再插入待验证设计的

[1] Register Transfer Level，寄存器传输级
[2] Field Programmable Gate Array，现场可编程门阵列

模拟仿真过程以实现采集和分析。覆盖率评估对于一般规模的集成电路设计（如普通单核超标量处理器）来说是一项十分成熟的技术，但对于复杂的片上多核处理器来说，覆盖率评估成了一个难题。虽然片上多核处理器的复杂共享存储和片上互连结构使得定义新的功能覆盖点和覆盖率模型非常必要且工作量巨大，但这些也只是一个经验、耐心和工作量问题，并不是片上多核处理器覆盖率评估困难的关键。对于模拟仿真来说，片上多核处理器覆盖率评估的难点在于前面提到过的模拟仿真速度问题——附加了覆盖率采集的模拟仿真对比原始模拟仿真速度成数量级下降。以功能覆盖率为例，普通超标量处理器定义覆盖率功能模型约 $10^3 \sim 10^4$ 个，大约会使得模拟仿真速度下降 2 个数量级，片上多核处理器具有更大的规模和更复杂的结构，定义的覆盖率模型数量将远超普通超标量处理器，导致本已难以容忍的模拟仿真速度问题“雪上加霜”。

既然基于（软件）模拟仿真的覆盖率采集遇到了困难，人们很自然地将希望寄托在片上多核处理器中广泛应用的**硬件仿真加速器**上，看起来与传统（软件）模拟仿真工具界面相似的硬件仿真加速器是否可以支持传统的覆盖率采集呢？首先我们需要分析一下硬件仿真加速器的基本工作原理。基于硬件模拟引擎的方法称为仿真加速的间接实现方法，与（软件）模拟仿真方法相似，采用基于周期和基于事件两种基本的模拟算法，所不同的是算法不是直接由软件实现，而是基于逻辑处理器网络实现。基于可编程逻辑的方法称为仿真加速的直接实现方法，直接将待验证设计映射到硬件的逻辑结构——可编程逻辑阵列(PLA)、可编程逻辑设备（PLD）或现场可编程门阵列。从上述两种硬件仿真加速方法的工作原理可以看出，基于硬件模拟引擎的方法从实现原理上与（软件）模拟仿真类似，最有可能支持覆盖率评估，而基于可编程逻辑的方法待验证设计经过综合和硬件映射后打乱了 RTL 原有的结构，使得基于代码和高层结构框架的覆盖率评估难以实现。目前市场上硬件仿真加速器基本分为基于现场可编程门阵列和门阵列/硬件模拟引擎二者结合两种，且前者由于价格相对便宜而市场占有率更高，从而使得基于仿真加速器的覆盖率评估难以实现。此外，与软件模拟相比，硬件仿真加速器速度快但牺牲了部分可调试性，常常要求设计，特别是测试平台设计，具有可综合性，且本身价格比较昂贵，通常只用于大规模集成电路设计的系统级验证，不能完全代替软件模拟仿真的功能。

由于硬件仿真加速器过于昂贵的价格，在硬件仿真加速器普及之前，传统超标量处理器设计多采用**现场可编程门阵列原型**来加速验证。现场可编程门阵列验证的缺点是发现设计错误后在现场可编程门阵列上调试非常困难，错误难以定位；优点是非常快的仿真速度，可以达到软件模拟的 1000 倍以上，可以工作在实际的目标环境中，可以在设计的早期进行系统级设计的验证，同时可以进行早期的性能评估和软件开发。然而，现场可编程门阵列方法是否能应用于片上多核处理器验证直接取决于设计的规模。以龙芯处理器为例：龙芯 2 号设计规模为 2640 万个晶体管，可勉强烧进当时能买到的最大容量的 FPGA Xilinx Virtex5；龙芯 3 号的处理器核采用裁减后的龙芯 2 结构，双核结构需要 2 片增强型 Xilinx Virtex5；龙芯 3A 为四核结构，考虑到复杂的片间互连问题，容纳该设计大约需要 3~4 片目前最大容量的 Xilinx Virtex6。由于现场可编程门阵列互连管脚和面积限制，4 片现场可编程门阵列划片已十分困难，6 片则几乎不可行。也就是说，4 核以上的龙芯 3 号设计就很难进行全系统规模的现场可编程门阵列验证了。事实上，虽然现场可编程门阵列领域本身的发展一直没有停止，各个现场可编程门阵列提供商都在努力提供更大容量的现场可编程门阵列平台和更高效率的现场可编程门阵列软件，但与集成电路设计规模的增长相比，始终难以满足大型集成电路设计系统级验证的容量需求。

无论是软件模拟还是硬件仿真加速、现场可编程门阵列原型验证都需要运行大量的测试向量。**测试向量生成**始终是需要重点关注的问题。测试向量来源可以是实际应用程序或专门

书写或随机产生的验证程序。测试向量的随机生成是处理器仿真验证技术发展过程中重要一步，使验证工程师不必总是面临需要搜肠刮肚书写更多的验证程序或为了一个小小的改动不得不把冗长的实际应用程序再次仿真一遍的窘境。随机测试向量生成技术最早起源于上世纪八十年代，经过研究人员和验证工程师二十多年的不断努力，随机测试向量生成技术经历了三个阶段的发展。目前，对于普通的单核处理器来说，随机测试向量生成技术已十分成熟。2004 年 IBM 公司的阿迪尔（A. Adir）等人建立的随机程序生成器 GenesysPro 是第三代随机测试向量生成技术的典型工具之一[9]。GenesysPro 主要对原始随机程序生成器的三个主要部分进行了改进创新：首先是测试模板的开发语言，通过使用该语言验证人员可以描述出复杂的程序结构，控制测试程序的生成行为；第二个改进是模型化的处理器结构，在 GenesysPro 中对于处理器模型化方法的优化提供了高度抽象的多种可配置模块模型，通过使用这些模型验证人员可以方便地生成被验证处理器的基本结构模型，并通过该模型在测试生成过程中指导程序的产生；改进的第三部分为随机约束生成引擎（CSP）。GenesysPro 是有文献报道的目前工业界比较成功的随机程序生成器之一。在传统随机测试向量生成技术领域，国内国防科技大学、清华大学、北京大学、浙江大学等单位以及中科院计算所也取得了一些研究成果。国防科技大学和北京大学就随机测试向量生成技术发表了一系列有价值的学术成果；清华大学的随机测试生成器已申请专利一项；中科院计算所 2005 年研制的指令级随机验证平台在一系列关键技术上取得了较好的进展，实现的指令级随机验证平台比 IBM Genesys 达到相同覆盖率目标的收敛时间缩短 30%左右，拥有与 GenesysPro 完备程度近似的功能，已经成为龙芯系列处理器验证的标准平台。

即使对于传统的超标量处理器，大量重复覆盖的测试向量造成的验证效率降低和资源浪费也是不可忽视的。近年来，减少重复测试向量，尽量用最少的测试向量达到最好的验证覆盖率，加速功能验证收敛成为模拟仿真验证方法的另一个重要研究方向。

通过覆盖率反馈驱动测试向量生成，减少重复测试向量，加速功能验证收敛是优化模拟仿真验证方法的重要途径。相关研究主要围绕三种思路进行[10]。意大利都灵理工大学（Politecnico di Torino, Italy）的科尔诺（Fulvio Corno） 等将基因演化算法应用于上述反馈过程，提出了测试向量演化（test program evolution）方法，基本策略是提出一个小的初始测试向量集，通过覆盖率反馈指导的一系列演化得到优化的测试向量集。该方法相对于人工的反馈过程是一个明显的进步，但其最大的难点在于初始测试集的选择。为了解决自动反馈问题，加州大学欧文分校（University of California，Irvine）的米施拉（P. Mishra）给出了另一种思路，先把待验证的处理器微体系结构的功能点抽取出来建立一个基于图的设计规范（Graph-based Specification），再根据设计规范建立覆盖率模型，将基于图的设计规范作为输入，采用形式化方法遍历设计规范，得到一系列的测试向量。由于覆盖率模型也来自同一个基于图的设计规范，很容易建立覆盖率模型与测试向量之间的对应关系，实现“全覆盖”验证。该方法最大的问题来自于处理器微体系结构功能点的抽取和覆盖率模型的建立。如果抽取简单的功能点建立模型，例如米施拉所采用的处理器流水线功能模型，该方法可以产生一定效果，但是如果抽取功能点过多则会造成模型复杂，形式化的遍历方法直接面临“状态爆炸”等一系列传统难题。费恩（S.Fine）等提出把贝叶斯网络技术应用在覆盖率动态反馈上，在基于算术指令的简单模型上取得了一定效果，但对于全相连的复杂状态覆盖率转移模型来说，由于覆盖率模型、待验证处理器属性点和生成的测试向量之间不是一一对应的关系，而是复杂的网状联系，需要建立更为庞大的推理库，高昂的实现代价限制了该方法的可行性和效果。显然，这三种思路都未能真正解决覆盖率反馈驱动测试向量生成的问题。此外，以形式化覆盖率引擎反馈驱动测试向量生成，达到加速验证收敛的目的也是一种新颖的思路。无论上述方法有着怎样的限制，实验结果都显示了这些方法对超标量处理器的随机验证改进

有一定作用。

现有覆盖率驱动的测试向量生成技术几乎都依赖于覆盖率度量技术和模拟仿真中待验证设计的白盒子结构。当处理器设计规模更大、结构更加复杂时，测试向量生成规则、待验证设计结构、覆盖率模型之间的对应关系会变得极为复杂，即使采用最先进的机器学习算法也难以处理这种复杂的三维网状关系。为了进一步简化问题，摆脱覆盖率驱动的测试向量生成技术对待验证设计结构细节的依赖，居泽伊（O. Guzey）和郭崎等分别提出了测试向量过滤技术[11,12]，将传统的在测试向量生成时进行的测试向量精简工作挪到测试向量生成后进行，采用支持向量机进行测试向量过滤。考虑到大规模设计模拟仿真时测试向量的生成时间远远小于其运行时间，该方法虽然增加了部分测试向量生成时间，还是有效克服了传统覆盖率驱动的测试向量生成技术在应用时遇到的难题。

前面介绍了测试向量生成技术相关的一系列研究成果，那么，当面对复杂的片上多核处理器时，这些研究成果是否还适用呢？首先来看传统的已经十分成熟的随机测试向量生成技术。片上多核处理器与普通超标量处理器相比不仅仅是设计规模增大了，还存在着结构上的创新，在如何为同构/异构多核、多线程及其复杂的共享存储结构生成各种复杂的验证程序以及复杂共享存储程序的正确性验证方面，需要巨大的改造和创新。再来看覆盖率反馈驱动测试向量生成技术。如果说测试向量精简对于普通单核处理器设计验证十分重要的话，对于设计规模更大、模拟仿真速度更慢的片上多核处理器来说，冗余测试向量问题将变得无法容忍，测试向量精简工作将变得更加必不可少。遗憾的是，片上多核处理器（软件）模拟仿真速度慢以及硬件加速仿真（或现场可编程门阵列原型）方法待验证设计综合后代码结构不可见等问题给覆盖率反馈 的应用带来了极大的困难，进而直接约束了覆盖率驱动的测试生成技术、甚至测试向量过滤技术的应用。要解决该问题，未来需要在包括片上多核处理器覆盖率度量、待验证设计划分剪裁、测试向量多层反馈生成、测试向量过滤等一系列关键技术上实现突破性创新。

除了测试向量生成外，基于模拟仿真验证的另一个关键环节是**正确性检测**。常用的正确性检测方法包括直接结果比较、记分板（Scoreboard）技术、自检测（Self-check）技术、运行后正确性检测技术等。对于复杂处理器设计来说，结果正确性检测技术通常离不开参考模型建立技术的支持。用于结果比较的黄金参考模型有许多种，包括指令集参考模型、体系结构参考模型、复杂处理器节拍精确的模拟器等。这些参考模型的复杂度差别很大，在实际验证中具体采用哪种参考模型还需要考虑黄金参考模型自身的正确性检测复杂度问题。对于大规模片上多核处理器验证来说，建立过于复杂的参考模型意味着需要把大量时间花费在参考模型自身的检验上。当参考模型复杂到一定程度，其验证复杂度已与待验证设计的验证复杂度相差无几，而过于简单的参考模型则可能丢失一些重要的必须检测的设计属性。在片上多核处理器实际验证中，要根据不同的验证需求在简单易检验的模型和复杂模型之间进行折中。然而，对于片上多核处理器的正确性检测来说，参考模型建立问题并非核心难题。真正的难题在于共享存储所导致的结果不确定性问题。存在数据竞争的测试程序在共享存储多处理器中执行结果的正确性由存储一致性模型定义。通常的存储一致性模型包括：顺序一致性（Sequential Consistency，SC)、全存储顺序（Total Store Order，TSO) 、松弛存储（Relaxed Memory Ordering，RMO)和释放一致性（Release Consistency，RC)等。现在已知：常用的共享存储模型验证执行的正确性问题为 NP 完全问题，这直接导致片上多核处理器结果正确性检测问题的计算复杂性。目前已有大量的研究围绕如何将检测执行正确性问题变为多项式或其它实际可接受时间内完成的问题进行，后面我们将结合片上多核处理器验证实例详细讨论这一问题。

模拟仿真验证的主要缺点是非完备性，即只能发现设计中的错误，而不能证明设计没有缺陷。**形式验证**通过数学方法证明待验证设计满足规范，它隐含考虑了所有输入向量的序列，能够完全证明设计是否符合规范，因而已经受到越来越多地关注。常用的形式验证技术包括：定理证明、等价性检查、模型检测、符号轨迹评估等[13~46]。

定理证明采用数学方法证明大规模集成电路设计的正确性，采用机械过程取代手工劳动对数学问题进行证明。定理证明的普适性非常好，不同类型、不同规模、有限域或无限域的系统都可以使用定理证明进行处理。但是其最大的不足在于自动化程度低下，虽然通过使用策略和决策过程可以在一定程度上提高自动化程度，但是人为的指导依然是必须的，而且占有相当大的比重。这意味着该方法的运用不仅需要定理证明工具设计者对数理逻辑有深入的了解，而且往往还要求定理证明工具的使用者准确控制程序的数学证明过程，无法完全自动化。这也正是到目前为止，定理证明只在一小部分领域得到应用，而在工业界无法得到广泛应用的一个主要原因。目前常见的定理证明器有 HOL、PVS、ACL2、Isabelle 等。

无论是模型检测还是等价性检查都离不开二叉决策图（Binary Decision Diagram，BDD）和命题逻辑可满足性问题（propositional satisfiability，SAT）理论的支持。二叉决策图以紧凑并且规范的形式表示布尔函数，经历了李（C. Y. Lee）、阿克斯（S. B. Akers）以及布莱恩特（R. E. Bryant）等人的一系列工作最终成型。二叉决策图有体积小、操作速度快、规范等特点，常被用于支持等价性检查和模型检测，本身也可支持组合电路验证。二叉决策图的乘法操作会产生空间爆炸，为此，人们研究了字级决策图（word-level decision diagram）。有别于二叉决策图，字级决策图表示基于布尔变量的整数函数，例如：MTBDD（Multi-Terminal BDD，多终端二叉决策图）或称 ADD（Algebraic Decision Diagram，代数决策图）、BMD（Binary Moment Diagram，二叉矩量图）、*BMD（Multiplicative BMD，乘法二叉矩量图）、HDD（Hybrid Decision Diagram，混合决策图）、*PHDD（Multiplicative Power HDD，乘法幂混合决策图）等。大多数字级决策图可以有效地处理乘法运算。命题逻辑可满足性问题简称 SAT 问题，是指给定一个命题逻辑表达式，判断是否存在一组变量赋值使该表达式为真。如果存在，则称表达式是可满足的；否则称表达式是不可满足的。很多电路验证问题都可以归结为 SAT 问题。虽然 SAT 问题是一个经典的 NP 完全问题，但很多实际的 SAT 问题仍可能存在高效的求解算法，例如：目前广泛使用的 DPLL 算法。

等价性检查验证两个设计是否实现了相同规范的所有操作或行为，是到目前为止应用得最成功的形式验证技术。目前等价性检查工具已广泛集成在大规模集成电路电子设计自动化流程中，保证超大规模集成电路设计流程中的频繁设计转换不会影响设计的正确性。然而，等价性检查只用于比较设计的一致性，对于证明设计本身的正确性无能为力。

模型检测是一种基于自动机的形式验证方法。系统被看作一个自动机，规范通常涉及系统在时间上的性质，因此以时序逻辑描述。如果自动机的行为符合规范，那么验证成功；否则模型检测器将给出一个反例，即自动机中的一条路径，以供用户分析。这种方法自动化程度较高，更有可能整合到工业界设计流程中，是近年来受到较多关注的技术之一。模型检测可以分为显式模型检测和符号模型检测，后者使用二叉决策图隐式表示自动机，从而扩大处理规模。此外还有基于 SAT 的有界模型检测（Bounded Model Checking，BMC），顾名思义，该技术只能检查自动机有限步骤内的性质，但因此可以获得更大的处理规模。模型检测是 NP 难问题，虽然近年来模型检测的处理能力有了较大的提高，但是这些方法依然有着指数级的复杂度，可处理的电路规模依然有限。迄今还没有一个成熟的自动化的基于模型检测的形式验证方法能在广泛的复杂工业级设计上取得成功。之所以如此，除了前面所述的原因，还有一个关键因素：在电路超出模型检测工具能完全处理的范围时（时间或者空间超过限

制），模型检测工具无法提供一些有用信息，告诉验证人员已经验证了哪些部分，还有哪些部分需要验证。

符号轨迹评估（Symbolic Trajectory Evaluation，STE）始于文献[25,26]，该技术在传统模拟仿真验证的基础上，通过引入三值仿真和符号仿真的概念，形成一种形式验证方法。此后杨（音译，J. Yang）等人又提出了广义符号轨迹评估（Generalized STE，GSTE）[27,28]，使 STE 可以处理无限时间上的属性。STE 十分类似于符号模型检测，但是其优势在于，通过三值仿真中的 X 提供了抽象机制，以便扩大处理规模，同时符号仿真中的符号常量和符号变量又为抑制抽象的失真提供了手段。运用 STE 需要在抽象程度和处理规模之间权衡。

长期以来，尽管形式验证理论在不断发展，但跟不上集成电路工艺增长、设计规模和复杂度提高的脚步，“状态爆炸问题”的阴云始终笼罩着形式验证，约束着形式验证技术的大规模全自动工业级应用。鉴于形式验证工具的处理规模有限，对于包含数千万乃至上亿个晶体管的片上多核处理器芯片，用形式验证方法来对其完全验证是不可能的。但作为模拟仿真验证的重要补充，形式验证方法仍然可以在片上多核处理器验证过程中发挥重要作用，主要包括三个方面：处理器核的形式验证、片上网络形式验证、缓存（Cache）一致性协议验证。三个方面中，针对处理器核中运算模块的形式验证已较为成熟，但针对控制模块的验证仍是目前形式验证研究的薄弱环节，而对于复杂结构的片上多核处理器验证来说，针对控制模块的验证显然十分重要。当前各种形式化验证方法都未能在处理规模和自动化程度上取得双赢，采用形式验证方法进行处理器核级完整验证的实例并不多见，随着形式验证技术的发展，采用形式验证完成完整处理器核级验证应该是可以期望的。片上网络形式验证虽然近年来研究较多，但面对片上多核处理器规模的快速增长，片上网络需要连接的处理器核数目快速增长，对相关形式验证方法的可扩展性提出了挑战。缓存一致性协议验证开始很早，1992 年克拉克（E.M. Clarke）等就利用符号模型检验器（Symbolic Model Verifier，SMV）首次成功验证了 IEEE Futurebus+标准 896.1-1991 缓存一致性协议，此后也有不少成功的实例。然而，缓存一致性协议实现于片上网络之上，在具体设计中，两者通常相互关联形成一体，且随着多核系统的日益复杂，存储层次也有增加的趋势，多级缓存必然导致缓存一致性协议的复杂化，这些都给形式验证带来了新的挑战。

事实上，由于存在健壮性和可扩展性问题没有解决这样的致命伤，目前形式验证仍游离在工业界设计流程和方法学之外，只能用于一些特定情况。近年来，形式化方法和模拟仿真方法相结合的半形式验证成为验证技术发展的趋势。相关的研究主要围绕两种思路进行。学术界研究的热点围绕在处理器微体系结构的形式化参考模型建立技术上，重点是建立算法级形式设计规范，从功能正确性检查、覆盖率评估到测试向量产生都依赖规范进行。相应的，由于这种思路对算法级形式设计规范过于依赖，使得形式设计规范本身的正确性和完整性成为设计验证的瓶颈，同时存在抽象图的抽象层次对验证的效果影响过大的问题。工业界主要考虑如何利用现有仿真平台，将系统级随机仿真验证与模块级形式验证结合起来，达到更好的验证效果，如 Synopsys 的 Megallan，依托 VCS 仿真器，在随机仿真验证中插入符号仿真和基于 SAT 的边界模型检测，以期达到更好的覆盖率。该方法优势主要体现在与电子设计自动化厂商原有的仿真平台结合紧密，速度快，但仍存在通过抽象状态机提取的覆盖率状态标准单一的问题，同时，处理规模限制仍然是目前面临的主要问题。此外，电子设计自动化验证环境的通用性要求其工具对所有的 IC 设计具有普适性，不能只针对处理器验证。即工具不需要关心设计的内部结构究竟是块转接子卡、桥接器还是处理器，更不必关心参考模型的建立，只需提供断言（assertion）描述语言来辅助用户自己定义断言，进行正确性检查。因此在进行处理器验证时用户仍需要提供知识库与这些通用电子设计自动化环境结合。

当前，国际上片上多核处理器早已成为处理器市场的主流产品，国内的片上多核处理器研制也正在实现从实验室原型到市场产品的跨越。对于大规模量产芯片来说，可测性设计必不可少。片上多核处理器结构复杂，设计规模巨大，需要占用大量测试资源。片上可以用于测试的管脚和测试通道是有限的，这限制了扫描链数量，导致扫描链的长度迅速增加，结果使得测试时间变得难以忍受。为了满足片上多核处理器可测性设计要求，大量可测性技术的创新会进一步增加片上多核处理器设计的复杂度，从而进一步增加片上多核处理器的验证难度。

此外，为了解决片上多核处理器设计面临的另一个重要问题——降低功耗，设计人员从晶体管级到结构级，在各个设计层次应用各种手段和技术。目前低功耗技术的常用方法包括结构级低功耗设计、门控时钟（Clock Gating）、功耗门控（Power Gating）、时钟树的低功耗设计、分级的功耗状态及功耗管理等。这些所带来的设计复杂度的提高使本就困难的片上多核处理器验证雪上加霜。

2.2 硅后验证

在过去的 20 年间，集成电路硅前验证水平取得了长足的进展，然而，与快速发展的工艺水平和集成电路设计水平相比，仍然难以满足大规模复杂集成电路验证实践的需要，常常会有各种错误和问题“逃过”硅前验证进入制造环节。事实上，以片上多核处理器为代表的现代大规模复杂设计，无论是高性能全定制芯片、半定制芯片、专用集成电路（ASIC）芯片或者系统级芯片（SoC），都面临着因种种问题导致的芯片多次流片问题。Collett International Research 研究报告指出，37%专用集成电路芯片面临 2 次流片，其中 24%甚至需要 3 次以上的流片。另有报道称 71％的重复流片源于逻辑错误，其余形形色色的错误和问题则与设计边界条件、功耗、电压、温度等各种物理电气特性相关（甚至与物理设计中缓冲电路（buffer）的摆放位置相关）。面对这些情况，芯片设计者和验证人员所能做的就是尽量用最少的流片代价将错误阻截在硅后验证阶段。考虑到无论面对怎样的困难，在芯片交付用户时仍然必须保证其正确性，否则就要付出惨重的代价（非官方的评估认为奔腾 4 处理器的召回成本为 600 亿美元），看似昂贵的硅后验证比起错误逃逸造成的后续问题来说完全是小巫见大巫，因此，近两年来，硅后验证变得越来越重要[47~52]。

硅后验证通过在实际应用环境中运行一个或多个制造后芯片来验证各种操作条件下芯片行为的正确性，目标是保证不要使错误“逃逸”到应用领域（用户手中）。一般来说，硅后验证主要包括四个步骤：

(1). 通过运行测试程序来发现问题。这些测试程序从随机指令序列到终端用户应用（如操作系统、游戏、科学计算等）都有，各种大大小小的测试程序会一直运行，直到系统发生失效（如系统崩溃、段错误或例外等）；

(2). 从大的系统失效中将问题定位到一个较小的区域。例如：一个复杂处理器上的程序崩溃可以被定位为在某种应用工作流下指令调度器产生的错误，能够暴露出错误的测试激励——特别是错误暴露得到的出错点周围数十行代码非常重要。

(3). 发现问题的根源。例如，某错误可能是由于电源噪声导致的——电源噪声会减慢电路某一路径上的信号传输，导致在某种输入序列下一些设计模块输出错误。

(4). 修复或旁路问题。可以采用打补丁、电路编辑等方法进行修复，极端状况下需要采用新掩模重新流片。

从硅后验证的目标和实施步骤来看，硅后验证与硅前设计验证和制造测试有些相似之

处。然而事实上，硅后验证在验证环境、运行速度、可调试性、可观测性等方面都与后两种存在极大差异，阻碍了现有相关技术的直接应用。已有若干工业界报告称：由于现有技术难以应对复杂芯片硅后验证需求，硅后验证已经变得极为困难且代价昂贵。片上多核处理器不断增大的设计规模和日趋复杂的设计结构注定硅后验证将在其开发过程中扮演越来越重要的角色，并最终变得不可或缺。作为一门新兴的技术，尽管近两年硅后验证越来越受到人们的重视，迫于实际需要，IBM、英特尔、AMD、ARM 等公司及学术界都投入了不少精力探讨和研究相关技术，但总的来说，硅后验证研究目前仅处于起步阶段，在错误重放、系统可观测性与可调试性、覆盖率度量、芯片耐受性和修复机制等许多方面都存在着大量的问题亟待解决。

除上述外，国内也有一些学术机构从事片上多核处理器验证的研究，但关于片上多核处理器验证方面的文献不多。江南计算所、国防科技大学就片上多核处理器验证开展了大量的研究与实践工作，在测试向量生成、缓存一致性验证等方面取得了一系列研究成果；清华大学姚文斌等开发了一款简单的单片多处理器验证环境 S-CMP，验证环境同时承担结构设计开发和系统验证的功能[53]；北京大学在覆盖率驱动的测试向量生成方面开展了一系列研究工作，取得了不少研究成果；中科院计算所依托龙芯 3 号系列多核处理器设计全面开展了片上多核处理器验证研究，目前已在测试向量生成、存储一致性验证、可调试性设计等研究领域取得了一些研究成果[55, 56]；浙江大学也开发了多核系统级芯片软硬件协同验证平台[54]；中国科学技术大学苏州研究院就片上多核处理器的并行仿真开展了一些研究工作，研究成果已申请发明专利两项；此外，另有其他一些单位也在多核处理器研制和验证方面有所涉猎，限于篇幅，此处不再一一列举。

前面对验证领域国内外研究现状进行了概括，分析了现有技术应对片上多核处理器验证的局限性，下面将结合商用 CMP 处理器验证实例具体说明当前片上多核处理器验证的现状及挑战。

3 商用片上多核处理器验证实例及问题

商业化的片上多核处理器已经面世几年，片上包含的处理器核个数也从初始的双核发展为 8~16 核，芯片设计横跨全定制、半定制、专用集成电路、系统级芯片等多种流程。在片上多核处理器相关的公开参考文献中，很大一部分用于描述结构设计/物理设计（包括低功耗设计等）实现及性能分析，详细描述片上多核处理器验证的不多，大致可分为两类：一类是聚焦片上多核处理器验证中的某具体问题或技术，此类点式研究多集中在存储一致性验证上，近两年来，硅后验证相关研究也开始逐渐进入人们的视线；第二类是片上多核处理器验证实践综述，此类文献对于全面了解和把握片上多核处理器验证来说非常珍贵，可惜数量很少，虽是管中窥豹，也可见一“斑”。

存储一致性验证方面的研究可追溯到大规模并行处理（MPP）时代。存储一致性模型定义了存在数据竞争的程序在共享存储多处理器中执行结果是否正确的标准。原康柏（Compaq）公司的泰勒（S. Taylor）等描述了一种基于仿真的验证 Alpha[3]结构共享存储属性的方法，该方法通过直接的无环图建立存储访问顺序模型，用户可以通过对照原始的结构正确性定义验证实现的各个方面。该方法缺点在于处理器一致性模型对特殊属性实现的依赖性，需要为每次处理器改进建立新的规则，改进新的版本。此外，缺乏形式化证明所建立的存储顺序模型的正确性。没有模型正确性的保证，基于模型的仿真验证就会隐患重重，可能

[3] 康柏开发的一款微处理器

发生放过冲突甚至误报错误等各种状况。

现在已知，常用的共享存储模型验证执行的正确性问题为 NP 完全问题，因而更多的研究围绕如何将检测执行正确性问题变为多项式或其它实际可接受时间内完成的问题[57~76]。由于存储一致性结果分析技术并不完备，许多研究只标记可以证明的违反存储一致性模型的状况，其他情况结果则无法确定，通常都乐观地假设机器的执行结果是正确的。Sun 公司/斯坦福大学的马诺维特（C. Manovit）和杭格尔（S. Hangal）描述了一种实用的新算法，据说可以解决上述确定性问题。该分析算法仅依赖程序可见的结果，不需要额外的系统内部信息，且算法工具可以运行在真实的商业处理器上。其实该算法只是实现了精确性和运行时间的又一种折衷。杭格尔等设计了一种 TSOtool——一种在共享存储的片上多处理器上检查存储子系统行为的程序。TSOtool 采用了一个改进的多项式时间的算法，在采用全定序存储（Total Store Order，TSO)一致性模型的系统上运行具有数据竞争的伪随机测试程序，然后检查程序运行结果是否符合形式化的全定序存储规范。这种方法虽然不完备，但实际有效。IBM 的拜尔（D. G. Bair）等采用形式化方法验证存储一致性协议，同时为存储控制器搭建单独的单元级仿真验证环境。宾州大学的马丁（Milo M.K. Martin）则采用反向思维，从可验证性角度思考设计问题，提出为了能在多处理器系统验证中应用形式验证，多处理器系统应该具有两种退耦属性：(1). 强制一致（coherence）和强制存储一致性（consistency）模型脱钩；(2). 缓存一致性协议与互连结构脱钩。方法是不要依赖任何特殊互连顺序和同步属性。并进一步分析了在两类主要的缓存一致性协议——目录机制和监听协议，认为目录机制比监听协议更符合要求。指出尽管监听协议概念简单，听起来更诱人，但其实现不符合上述退藕要求，推理、验证和正确实现较困难。反之，目录机制看似复杂，但更符合退耦要求，简化了协议的设计和验证。目前只有少部分设计采用了上述退耦规则，作者倡议在设计时考虑可验证性需求，认为这样验证和设计都会事半功倍。然而，考虑到影响处理器结构设计的因素很多，例如物理设计水平等，给结构设计增加了很多限制，上述倡议看来难以实现[64]。2009 年，陈云霁等提出在片上多核处理系统中访存操作之间存在一种自然序关系，称为“时间序”。利用时间序，可以在不损失精确性的情况下有效地将片上多核处理器存储一致性验证问题局部化，将存储一致性验证的复杂度降为 $\boldsymbol{O}(\boldsymbol{p}^3\boldsymbol{n})$，引入微结构相关知识进而将复杂度降低为 $\boldsymbol{O}(\boldsymbol{n})$，实现线性复杂度的片上多核处理器存储一致性验证[56]。此外，谢赫（A. A.Sheikh）等也简单描述了基于系统级芯片的异构多处理器系统的系统级验证方法。

硅后验证作为新兴的技术领域近来得到了广泛的关注和重视，2010 年电子设计自动化领域的国际顶级会议设计自动化会议（Design Automation Conference， DAC）上就该领域邀请来自 IBM、英特尔、Qualcomm 、ARM 等验证团队的负责人、电子设计自动化业界专家与世界各地从事验证研究的学者展开了深入讨论，表现了工业界和学术界对硅后验证领域的极大重视。

在商用处理器领域，2007 年，IBM、英特尔和 AMD 分别首次发表了关于硅后验证的文章。有趣的是，三篇文章都强调了多核处理器，这从一个侧面佐证了多核处理器加剧了“验证危机”并直接推动了硅后验证的兴起[47~52]。IBM 重点描述了其设计的一款异构多核处理器芯片 IBM Cell/B.E 以及多款游戏机处理器芯片的硅后验证情况，就测试用例的生成、缓存一致性、 TLB[4]一致性、软件管理 TLB 等方面应用具体实例进行了非常简单的介绍；英特尔描述了其双核处理器 Core2 Duo 的硅后验证方法，重点介绍了系统验证、兼容性验证以及代工厂商（OEM）联合验证三个方面；AMD 则将硅后验证划分为三个阶段：初始阶段主要执行基础诊断和加载操作系统，中间阶段会运行各种各样的平台以及相应的软件以验证处

[4] 翻译后援存储器，Translation Lookaside Buffer

理器的功能和兼容性，最后阶段则主要关心内存一致性、内存排序、缓存一致性和缓存原子性。文中还描述了硅后错误调试的过程。2010 年，ARM 公司首次表达了对硅后验证的重视，发表了题为“硅后验证已开始得太晚—通过设计验证避免生产 5000 万美元的镇纸”的文章[51]。ARM 是国际上市场占有率最高的集成电路 IP 核提供商。IP 核通常分为软核、硬核和固核三种。这决定了 ARM 公司的硅后验证更像硅前验证的延续，其产品对低功耗设计的极度重视也使得复杂低功耗设计验证成为其硅后验证的重点。总的来说，目前商用处理器关于硅后验证的文献很少且较为浅显，并未触及硅后验证领域如错误重放、系统可观测性与可调试性、覆盖率度量、芯片耐受性和修复机制等许多核心问题，而这些问题仍是横亘在硅后验证面前的重要挑战。

近年来，虽然著名的处理器厂商英特尔、IBM、AMD 等纷纷推出自己的多核处理器，但关于片上多核处理器验证实践综述类的描述却较少见诸文献，只有 IBM 粗略描述了 Power 系列多核处理器验证实践[5, 77~79]，以及 NEC 从验证管理角度对嵌入式多处理器验证稍有涉及，英特尔等处理器厂商多三缄其口。

NEC 公司面向高性能、低功耗的消费电子领域，设计了一款片上多处理器原型—Merlot 来证明其猜测多线程结构。Merlot 集成了片上设备、PCI[5]接口和 SDRAM[6]接口，芯片设计涉及目前的片上多处理器和系统级芯片设计领域。文献从工业化验证管理的角度介绍了 Merlot 的验证，详细描述了 Merlot 设计流程、验证环境、验证层次和验证流程，指出严格的设计规则和验证流程对保证验证成功的重要性。

IBM Power 系列处理器及其系统描述如表 2 所示（该表中尚未包括 IBM 最新的 Power7、8 核处理器，后面将对其验证进行单独介绍）

表2. IBM Power 系列处理器及其系统描述[79]

		描 述	规格说明页数
系统	Power4+	基于 Power 的高端服务器	~1260
	Power5	更多处理器 多线程 微分割 新读写芯片 增强型存储子系统	~2600
	Power5+	增强型 Power5 系统	~2600
	Power6	Power5+的增强型	~2600
	Cell	Power 架构的增强	~1500
处理器	Power4+	高端 Power 处理器	~710
	Power5	多线程 微分割	~770
	Power5+	Power5 的增强型	~780
	P-X	一款 64 位 Power 芯片	~740
	Power6	对 P ower5+的强型	~770
ASIC 芯片	Bridge#1	PCI 桥	~290
	Bridge#2	主机总线，更多 PCI 端口	~400

Cell Broadband Engine(Cell BE)是由索尼、东芝、IBM 联合开发的。Cell BE 是以 IBM 研发的 64 位 POWER 为核心（PPE），结合 8 个互相协作的处理机单元（SPEs）构成的处理器。CEBA 架构可适用于多样规划设计，并且支持 PPE 和 8 个 SPEs 分离工作。

验证包含 2.34 亿个晶体管的 Cell Broadband Engine 涉及跨五个时区的五个参与公司的庞大团队，项目管理复杂，非常具有挑战性。在验证 Cell 的过程中，基于仿真的验证技术在验证过程中发挥了决定性作用，验证策略包括层次化验证(hierarchical verification)、随机验证(random verification)和基于跟踪的结

5 Peripheral Component Interconnect，互连外围设备
6 Synchronous Dynamic RAM，同步动态存储器

构验证(trace-based architectural verification)。随机验证环境包括一个输入端动态测试向量产生程序和一个输出端的动态检测程序。随机测试向量产生程序是系统级的 X-Gen。结果检测有两种方法：模块级的实时检测和芯片级的基于跟踪的验证。Cell 首次流片前共仿真了超过 2 万亿个节拍，流片回来后很快就能运行操作系统和一些应用。

POWER4 和 POWER5 都是用于 IBM 服务器系统的双核微处理器。POWER4 包括 150 万行 VHDL[7]代码和 1.74 亿晶体管，POWER5 包括 2.76 亿晶体管，结构复杂。

Power4 处理器包含两个 CPU 核，每个处理器核拥有 64KB 一级指令缓存与 32KB 一级数据缓存，两个核共享三个 512KB 二级缓存，第三级缓存采用 eDRAM 内存，容量从 32MB 到 128MB。对于 POWER4 的验证，IBM 公司可谓全力投入。除了将以往 POWER 系列总结和积累的验证工具加以改进外，还专门为 POWER4 的验证开发了新的验证工具。仅文献列举的已命名的验证工具就多达 15 种，包括仿真工具 TexSim/TexVHDL，硬件加速器 AWAN，等价性检查工具 Verity，功能形式验证工具 RuleBase，四个覆盖率工具 Bugspray、Comet、Abacus 及 Covet，模块级验证工具 TIMEDIAG 和 GENRAND，随机指令级测试用例生成工具 Genie、Genesys 和 GenesysPro，一致性检查工具 CML 等，在模块级、单元级、多单元级、芯片核级和系统级等多个层次进行验证。POWER 系列处理器积累了大约 50 万个常规测试集（不包括随机验证程序），其中 15 万个测试集在每一级模型和每次 VHDL 修改后的版本的回归测试中都要使用，而在较大的修改发布时必须完成全部测试集的回归测试。为了仿真如此大而复杂的处理器设计，IBM 动用了数千台 pSeries 工作站和服务器的机群，分别放在德克萨斯州的奥斯汀和明尼苏达州的罗彻斯特，每处机群都以 100M 以太网互连，仿真环境的平均处理能力为 10 亿节拍/天。尽管如此，POWER4 第一版 2000 年流片回来，在多种配置上成功运行 AIX 和 LINUX 以后，还是发现了一些漏网（bug)，只好通过软件或性能降级把这些错误绕过去。

Power5 是 IBM 推出的双核同时多线程处理器，集成了两个处理器核，每个核为同时多线程（Simultaneous Multithreading，SMT）处理器，能够同时执行两个线程，片内集成了 1.92MB 的二级缓存，此外还集成了三级缓存的目录及存储控制器。因为 POWER5 是对 POWER4 的向上兼容，因此对 POWER4 的验证方法全部被移植用来应对 POWER5 的验证。除此以外，还针对 POWER5 的验证提出了很多新工具和技术改进：包括同时多线程增强型测试用例产生工具 MultiProcessor Test Generator (MPTG)、系统级测试用例加载、执行、规则检查、结果检验工具 SPECTOR、读写测试用例产生工具 X-Gen、增强型测试覆盖率分析工具 SixthSense、覆盖率分析数据库 Meteor 系统并配合半形式化的验证工具 SixthSense 对 Genesys-Pro 进行了改进。

由于规模更加庞大，对 Power5 的验证在方法论上有了重大变化，引入了验证优先级的概念。与 Power 4 相比，Power 5 具有更复杂的结构，但是也重用了 Power 4 的设计。因此在 Power 5 的验证过程中采用新特性优先法则，保证 Power 5 的新特性得到更深入的验证。并且从 Power 4 采用大量的随机产生的测试用例转向 Power 5 中的以验证事件为导向，每个验证事件都有特定的测试目的，所有与 Power 5 相关的测试事件被标记为最高优先级。验证人员、具体实现设计人员和体系结构设计人员共同决定哪些 Power 5 的功能单元需要更高的验证测试覆盖率。共计为 Power 5 的验证定义了超过 35,000 个测试用例以覆盖验证事件。在测试用例的运行过程中，验证系统不间断地搜集统计数据，排除无效的测试用例以提高验证的效率，并保证最高优先级的测试事件得到持续的覆盖。Power5 的验证团队保证了首次流片支持操作系统(AIX, Linux, i5/OS)启动，并在流片之前发现了 95%的设计问题，只剩下不

[7] VHSIC hardware description language，超高速集成电路硬件描述语言

到 1%的设计问题会影响芯片的使用。验证团队进一步通过软件改进等其它方法提供了对于这些设计问题的解决方案。

Power7 是 IBM 第一款高端 8 核处理器。与此前 IBM Power 系列产品相比，Power7 处理器不是处理器核的简单堆砌，而是有许多重大的结构创新，首次实现了共享片上三级缓存，缓存基于嵌入 DRAM[8]单元，而非 SRAM，并被集成到芯片中。芯片在 567 mm^2 上集成了 12 亿晶体管，八个处理器核采用 4 路对称多线程（symmetric multithreading，SMT）结构连接，设计了私有二级缓存及前述共享三级缓存，为 SMP 连接提供了 7 个连接端口，两个内存控制器可传输 100 GB/s 的数据，还为远程访存（Remote Access Service， RAS）和电源管理提供了附加功能。Power7 的一系列结构创新导致片上晶体管数量急剧增加。尽管增加的晶体管大部分用于增加缓存大小以及提高多核并行性，但仍有大量晶体管用于开发更复杂的功能，从而加剧了“状态空间爆炸”现象，而其多锁相环（PLL）和组件间多种多样的异步接口设计更使业已十分困难的验证更加困难。尽管 IBM 在完成此前一系列多核产品（如前述 Cell、Power4、Power5 等）验证的过程中积累了大量经验，开发了多达几十种的验证工具，仍不足以应付复杂的 Power7 验证工作。

恶化的“状态空间爆炸”问题、难以精确预测的复杂的行为设计以及有限的验证时间和资源使得 Power7 验证成为 IBM 系列处理器验证中难度最高的芯片之一。IBM 通过改进测试向量生成技术、结构检查技术以及形式验证技术等来缓解“状态空间爆炸”问题，应对多核多线程处理器的验证需求。然而多核多线程异构测试向量的复杂交互仍是目前测试向量生成技术的难点所在。为此而采取的对策有：改进结构检查技术，开发各种点式工具，以便不通过模拟仿真即可验证设计的某些属性，节省有限的仿真资源和验证时间；改进形式验证技术，除传统的运算电路形式验证外，加大协议验证、特别是片上网络形式验证的力度，这就同时简化了时序电路等价性检查流程；采用一些技术改进和折中，缓解共享存储、猜测执行、低功耗复杂电源管理措施等引起的结果预测检查的困难。现代复杂处理器设计中的一些功能属性，如错误修复机制、功耗管理中的处理器部分“休眠状态”等非常难以检测，往往需要长指令序列，花费数以十万计的时钟周期才能充分驱动相应的硬件机制。因此，设计复杂的长指令序列非常重要。然而，一些多种机制交互的情况十分难以覆盖。例如，复杂电源管理功能和高速缓存一致性交互时出现的一些“诡异”的窗口问题或边角情况，指望采用指令序列覆盖基本是不可能完成的任务。验证流程离不开覆盖率度量，Power7 验证中大量运用了硬件仿真加速器，这对覆盖率采集分析技术提出了挑战：怎样才能在硬件仿真加速器上完成覆盖率度量？怎样才能减少覆盖率采集对仿真加速器性能的影响？长时间采集的覆盖率数据怎样存储和管理？这些问题都有待回答。此外，除了技术挑战，验证项目管理问题也已经成为 Power7 验证过程中的难关。多个地域、多个项目、多个工作小组的协调与交互所需要的高度复杂的信息管理能力已成为制约验证效率的重要因素。与 Power4、Power5 验证不同，IBM 在介绍 Power7 验证工作时不再将重点放在工具开发和具体技术分析上，而是更多地介绍了 Power7 验证工作中碰到的一些挑战和问题，其中的一些内容甚至仅仅只是“问题”，在文中根本找不到答案。这其实并不奇怪，片上多核处理器验证中的许多问题目前仍是当前整个验证领域尚未解决的，而“Power7 验证挑战”一文中只是离散地列出了实际验证工作中碰到的问题的一小部分，只是“冰山一角”。

综上，尽管涉及商用片上多核处理器验证的参考文献很少，但对仅有的几篇参考文献进行分析，仍不难在字里行间发现验证工程师在面对大规模片上多核处理器验证时所面临的困难和无奈。首先是流片后的错误率问题。普通超标量处理器验证时难得一见的流片后错误在

[8] Dynamic Random Access Memory，动态随机存取存储器

片上多核处理器验证时变得十分常见，原因在于片上多核处理器模拟仿真速度极慢，对于不惜大成本资源投入的大的处理器厂商来说，IBM 仿真环境的平均处理能力所达到的 10 亿节拍/天的水平几乎到了成本和能力所能支持的极限。处理器芯片设计不可能提供无限的时间用来验证。假设一个芯片的验证周期是一年，则可运行 3650 亿节拍仿真向量，而这仅相当于实际芯片运行几十分钟。对于一个商业芯片产品来说，不经过 2~3 个月的硅後验证是不敢拿给用户使用的，从这种巨大的落差可以看出片上多核处理器验证面临流片后的错误率问题并不奇怪。其次是验证过程和质量的度量问题。覆盖率是指导验证过程和衡量验证质量的重要手段。英特尔的一篇文献曾描述了该公司为验证一款单核商业芯片所做的功能覆盖率方面的巨大工作量，共定义了上千万个功能覆盖点，花费了大量的人力物力用于覆盖率的采集和分析，最终仍有大量的功能覆盖点难以覆盖且分析困难，只好不了了之。不难想象，对于片上多核处理器这样规模更大、复杂度更高、模拟仿真速度更慢的芯片来说，覆盖率工作必然困难重重，进而会影响到整个验证过程的收敛速度和质量。

前述 Power4、Power5、Power7 的验证只是当前多核商用处理器验证的一个缩影。可以合理的推论，包括英特尔、AMD 在内的其它处理器厂商也同样会花费很大的力量来验证其多核产品，也会面临相似的困难。随着集成电路工艺的发展，片上多核处理器结构设计也呈现出越来越复杂的趋势，所包含的片上处理器核数目从 2、4、8 一路增加到 64 核。片上多核处理器验证尚未解决老的大规模设计仿真速度慢、硬件加速规模不足和形式验证处理能力不足的问题，又面临新的存储一致性和复杂片上互联等方面的验证困难。国外大处理器厂商尚且如此，面对同样的问题，国产片上多核处理器验证就更不容乐观了。

4 关于国内片上多核处理器验证现状及未来的思考

近年来，国家出于安全战略以及增强国际竞争力的考虑，对国产处理器研制工作非常支持。而对于国产高性能处理器芯片的研制者，片上多核处理器结构具有很强的吸引力：片上的处理器核可以很简单，使得设计时间短，也易于获得较高主频，同时与另一种高性能处理器结构—同时多线程结构相比，易于克服线延迟问题，具有明显的低功耗优势，可以在国内物理设计水平不高的情况下实现。这为国产高性能处理器研制迎来了前所未有的机遇，而片上多核处理器的研制也使後发的国产处理器研制基本赶上了国外处理器研制领域前进的脚步。目前国内已经有多家单位从事片上多核处理器结构的研究，国产的包含四核、八核、甚至十六核的片上多核处理器已经成功流片，这些都标志着国产处理器研制工作的巨大进展。然而，看到了进展，也要看到差距和问题。以前的国产处理器研究基本处于追着国外跑的状态，很多实现方面的问题可以找到借鉴和参考的对象。在开展片上多核处理器研制时，由于国内外差距的缩小，很多实现中的具体问题已经很难找到借鉴和参考的对象，验证技术作为处理器设计中最重要和繁琐的环节就首当其冲碰到了困难和瓶颈。与此同时，无论从国家安全战略、还是真正增强国际竞争力的目的出发，国产高性能处理器研制的最终目标并不是实验室的原型，而是真正能够使用和推广的商业产品。要实现这一目标，就绕不开验证技术。

以本文作者所在的龙芯团队为例，虽然经过龙芯 1 号、龙芯 2 号 7 个版本的考验，团队在系统仿真、现场可编程门阵列验证、硬件加速仿真、指令级随机验证、模块级比较验证、形式验证、覆盖率分析、断言检查等技术领域积累了不少经验，在应用程序仿真过程中也积累了不少测试用例集，同时还在指令级随机验证、多层次处理器建模、半形式化驱动覆盖率反馈、形式验证等领域提出和应用了很多新的技术和方法，但对于龙芯 3 号系列片上多核处理器的验证复杂度来说上述技术仍然不够用。在龙芯 3A（四核）和龙芯 3B（八核）片上多核处理器验证实践中碰到了很多新的困难和挑战。举例来说，以前的处理器验证非常依赖的

全系统级现场可编程门阵列仿真在片上多核理器验证时就不再适用，只能依靠新的验证方法和技术来弥补。而这只是传统验证方法局限性的一个缩影。

传统验证技术已经无法满足片上多核高性能处理器验证的需要，新的技术亟待开发和研究，这不只是国产片上多核处理器面临的问题，也是国际片上多核处理器研制共同面临的严重问题。国外大处理器厂商基本采用资源密集型验证方法，投入惊人的人力、物力资源来解决这些问题。问题的解决过程更是极少出现在公开的文献中。尽管如此，从仅有的几篇文献中还是不难看出它们在片上多处理器验证方面的巨大投入，甚至极其隐讳地暗示其在某些方面的无可奈何。例如英特尔曾经提及覆盖点设计过多时，覆盖率分析的困难重重甚至只好无奈放弃；IBM 则提到以前从未采用过的验证优先级的概念，把验证任务分了轻重缓急，且直接认可流片后的部分错误存在，把更大的压力留给硅后验证。对国内片上多核处理器开发者，采用大投入解决验证问题不现实，只有把更多的力气投入到开发先进的验证技术才是解决问题的现实可行的必经道路。

5 结束语

片上多核处理器是目前商用处理器的主流产品，也是最复杂的一类集成电路产品。随着集成电路工艺的发展，多核结构设计也呈现出越来越复杂的趋势。无论对于国际知名的处理器厂商还是国内新兴的处理器产业来说，多核处理器验证都是一项非常具有挑战性的工作。由于现有的验证技术无法应对和解决片上多核处理器验证中的困难和问题，近年来，大量的研究围绕多核处理器验证开展，主要包括以下几个方面：

针对模拟仿真验证的速度和处理规模问题，围绕高效测试向量生成技术、测试向量精简技术、层次化模拟仿真技术、硬件仿真加速等关键技术开展研究；

针对片上多核处理器正确性检测和验证品质度量问题，围绕覆盖率建立分析技术、多级参考模型建立技术、正确性检测技术等关键技术开展研究；

针对形式验证问题，围绕共享存储协议验证、片上互联验证以及验证规模拓展等方面开展研究；

针对硅后验证问题，围绕错误重放技术、可调试性设计技术、硅后芯片耐受性技术、以及低功耗设计验证技术等方面开展研究；

另有大量针对片上多核处理器验证中各个环节的点式关键技术研究及验证项目管理研究，限于篇幅不再一一列举。

目前，我国高性能处理器的研制和产业化都处于关键时期，片上多核处理器研制成为当前工作的重点。作为片上多核处理器研制最大的难点之一，验证已经成为高性能处理器研制的真正瓶颈，这对我国的广大验证工作者来说，既是一个巨大的挑战，又是一个难得的机遇。解决了这个问题，我国就有可能在处理器研制领域追赶上世界的脚步，而我国的处理器验证技术将会处于世界领先水平；失去这个机遇，我国的处理器研制水平与国际水平的差距将越来越远，处理器产业化进程也将遭遇极大困难。在缺乏像国外大公司那样长期的技术积累和雄厚的财力支撑的情况下，如何设计和验证具有竞争力的国产片上多核处理器？这是摆在我们面前的现实问题，也是我辈处理器验证工作者的重任。

参考文献：

[1] Hammond L., Nayfeb B.A., Olukontun K.. A single-chip multiprocessor. IEEE Computer, 1997, 30(9):79-85

[2] Hammond .L., Hubbert B., et al. The Stanford Hydra CMP, IEEE Micro, 2000, 20(2):71-84

[3] Kalla R., Sinharoy B., Tendler J.M.. IBM Power5 Chip: A dual-core multithreaded processor. IEEE Micro, 2004, 24(2):40-47

[4] Sinharoy, B.，POWER7 multi-core processor design ，2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on 2009 , Page(s): 1

[5] Schubert, K.-D.，POWER7 — Verification challenge of a multi-core processor，IEEE/ACM International Conference on Computer-Aided Design 2009. ICCAD , Page(s): 809 - 812

[6] Pham D., Behnen E., et al. The Design and Implementation of a First-Generation CELL Processor. In: Proc. of the 2005 IEEE International Solid State Circuits Conference (ISSCC), San Francisco, CA, USA, February 2005, 45-50

[7] Bob. Bentley, “Validating the Intel Pentium 4 Microprocessor”, Proc. 38th Design Automation Conf. (DAC 01), ACM Press, 2001, pp. 244-248.

[8] Nahir, A.; Ziv, A.; Abramovici, M.; Camilleri, A.; Galivanche, R.; Bentley, B.; Foster, H.; Hu, A.; Bertacco, V.; Kapoor, S.，Bridging pre-silicon verification and post-silicon validation，47th ACM/IEEE Design Automation Conference (DAC), 2010, Page(s): 94 - 95

[9] A. Adir et al. “Genesys-Pro: innovations in test program generation for functional processor verification”, IEEE Design & Test of Computers, , Vol.21 ,Issue: 2 , Mar-Apr 2004 ,pp.84 – 93

[10] 沈海华、卫文丽、陈云霁，“覆盖率驱动的随机测试生成技术综述”，《计算机辅助设计与图形学学报》，2009 Vol. 21 No.4 pp.419-431.

[11] Guzey, L.-C. Wang, J.Levitt, and H.Foster. Functional test selection based on unsupervised support vector analysis. In DAC'08: Proceedings of the 45th Design Automation Conference, Pages 262-267, 2008

[12] Q. Guo, T. Chen, H. Shen, Y. Chen, W. Hu. On-the-fly reduction of stimuli for functional verification. In ATS'10: Proceedings of the 19th Asian Test Symposium, Pages 448-454, 2010

[13] Jacobi, C.,et al, “Automatic formal verification of fused-multiply-add FPUs”, Design, Automation and Test in Europe(DATE05), 2005 Page(s):1298 - 1303 Vol. 2

[14] Ayewah, N.; Kikkeri, N.; Seidel, P., “Challenges in the Formal Verification of Complete State-of-the-Art Processors”, International Conference on Computer Design (ICCD2005), 02-05 Oct. 2005 Page(s):603 - 608

[15] Ganai, M.K.; Aziz, A.; Kuehlmann, A. “Enhancing simulation with BDDs and ATPG”, the 36th Design Automation Conference(DAC99), 21-25 June 1999 Page(s):385 - 390

[16] Biere, et al. “Symbolic Model Checking Using SAT Procedures Instead of BDDs”, the 36th Design Automation Conference(DAC99), 21-25 June 1999 Page(s):317 - 320

[17] P.Bjesse, et al. “Finding Bugs in an Alpha Microprocessor Using Satisfiability Solvers”, the 13th International Conference on Computer-Aided Verification, 2001

[18] Biere, et al. “Verifying Safety Properties of a PowerPC Microprocessor Using Symbolic Model Checking without BDDs”, the 11th International Conference on Computer Aided Design(ICCAD04),LNCS, vol.1633, 1999 Page(s):60 - 71

[19] Nina Amla, et al. “Experimental Analysis of Different Techniques for Bounded Model Checking”, the 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), volume 2619 of LNCS, pages 34¨C48. Springer, April 2003.

[20] Shimizu, K.; Dill, D.L. “Using formal specifications for functional validation of hardware designs”, IEEE Design & Test of Computers, Volume 19, Issue 4,July-Aug. 2002 Page(s):96 – 106

[21] Wen-Shiu Liao; Pao-Ann Hsiung, “FVP: a formal verification platform for SoC”, IEEE International SOC Conference, 17-20 Sept. 2003 Page(s):21 - 24

[22] Yunshan Zhu; Marshall, T. “Design verification using formal techniques”,the 4th International Conference on ASIC,23-25 Oct. 2001 Page(s):21 – 28

[23] Tasiran, S.; Yuan Yu; Batson, B. “Linking simulation with formal verification at a higher level”, IEEE Design & Test of Computers, Volume 21,? Issue 6,? Nov-Dec 2004 Page(s):472 - 482

[24] Rossi, U. “Can we really do without the support of formal methods in the verification of large designs?”, the 42nd Design Automation Conference(DAC2005), 13-17 June 2005 Page(s):672 – 673

[25] -J. H. Seger and R. E. Bryant. Formal Verification by Symbolic Evaluation of Partially-Ordered Trajectories. Formal Methods in System Design, 6(2), 1995. pp. 147 - 189.

[26] S. Hazelhurst and C. -J. H. Seger. A Simple Theorem Prover Based on Symbolic Trajectory Evaluation and OBDDs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 14(4), 1995. pp. 413 - 422.

[27] J. Yang and C. -J. H. Seger. Introduction to Generalized Symbolic Trajectory Evaluation. Proceedings of the International Conference on Computer Design, 2001. pp. 360 - 367.

[28] J. Yang and C. -J. H. Seger. Generalized Symbolic Trajectory Evaluation - Abstraction in Action. Proceedings of the International Conference on Formal Methods in Computer-Aided Design, 2002. pp. 70 - 87.

[29] Jun Sawada, Warren A. Hunt, Jr.. Verification of FM9801 - An Out-of-Order Microprocessor Model with Speculative Execution, Exceptions, and Program-Modifying Capability. Formal Methods in System Design, Volume 20 Issue 2, March 2002

[30] Matt Kaufmann, J. S. Moore. An industrial strength theorem prover for a logic based on common lisp. IEEE Transactions on Software Engineering, Volume 23 Issue 4, April 1997

[31] David M. Russinoff. A case study in formal verification of register-transfer logic with ACL2 - The floating point adder of the AMD Athlon processor. FMCAD, 2000

[32] Christian Jacobi, Christoph Berg. Formal Verification of the VAMP Floating Point Unit. Formal Methods in System Design, Volume 26 Issue 3, May 2005

[33] Roope Kaivola and Naren Narasimhan Formal Verification of the Pentium 4 Floating Point Multiplier DATE, 2002

[34] Christian Jacobi, Kai Weber, Viresh Paruthi, Jason Baumgartner Automatic formal verification of fused-multiply-add FPUs, DATE. 2005

[35] Adrian E. Seigler, Gary A. Van Huben, Hari Mony. Formal Verification of Partial Good Self-Test Fencing Structures. FMCAD, 2007

[36] Thuyen Le, Tilman Glökler, Jason Baumgartner Formal Verification of a Pervasive Interconnect Bus System in a High-Performance Microprocessor DATE, 2007

[37] Alon Flaisher, Alon Gluska, Eli Singerman Case study - Integrating FV and DV in the verification of the Intel Core 2 duo microprocessor , FMCAD, 2007

[38] Jin Yang; Ghughal, R.; Tiemeyer, A. Industrial scale formal verification using concurrent GSTE. ASICON 2005.

[39] Roope Kaivola. Formal Verification of Pentium 4 Components with Symbolic Simulation and Inductive Invariants. CAV, 2005.

[40] Manish Pandey, Randal E. Bryant. Exploiting symmetry when verifying transistor-level circuits by symbolic trajectory evaluation. IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 18, NO. 7, JULY 1999

[41] E. M. Clarke, O. Grumberg, and S. Jha, “Verifying parameterized networks,” ACM Transactions on

Programming Languages and Systems, vol. 19, no. 5, pp. 726–750, 1997.

[42] R. Bharadwaj, A. Felty, and F. Stomp, “Formalizing inductive proofs of network algorithms,” in Proceedings of the Asian Computing Science Conference on Algorithms, Concurrency and Knowledge (ACSC ’95), pp. 335–349, Pathumthani, Thailand, December 1995.

[43] H. Amjad, “Model checking the AMBA protocol in HOL,” Tech. Rep., Computer Laboratory, University of Cambridge, Cambridge, UK, September 2004.

[44] Gebremichael, F. W. Vaandrager, M. Zhang, K. Goossens, E. Rijpkema, and A. Radulescu, “Deadlock prevention in the Æthereal protocol,” in Proceedings of the 13th IFIP WG 10.5 Advanced Research Working Conference Correct Hardware Design and Verification Methods (CHARME ’05), pp. 345–348, Saarbr¨ucken, Germany, October 2005.

[45] Herzberg and M. Broy, “Modeling layered distributed communication systems,” Formal Aspects of Computing, vol.17, no. 1, pp. 1–18, 2005.

[46] Schmaltz and D. Borrione. Towards a Formal Theory of On Chip Communications in the ACL2 Logic. In Proceedings of the Sixth International Workshop on the ACL2 Theorem Prover and Its Application, pages 47-56. ACM, August 2006.

[47] DeOrio, A.; Bauserman, A.; Bertacco, V.; , "Post-silicon verification for cache coherence," Computer Design, 2008. ICCD 2008. IEEE International Conference on , vol., no., pp.348-355, 12-15 Oct. 2008

[48] Keshava, J.; Hakim, N.; Prudvi, C.，Post-silicon validation challenges: How EDA and academia can help，47th ACM/IEEE Design Automation Conference (DAC), 2010, Page(s): 3 - 7

[49] Mitra, Subhasish; Seshia, Sanjit A.，Post-silicon validation opportunities, challenges and recent advances，47th ACM/IEEE Design Automation Conference (DAC), 2010 , Page(s): 12 - 17

[50] Nahir, A.; Ziv, A.; Abramovici, M.; Camilleri, A.; Galivanche, R.; Bentley, B.; Foster, H.; Hu, A.; Bertacco, V.; Kapoor, S.，Bridging pre-silicon verification and post-silicon validation，47th ACM/IEEE Design Automation Conference (DAC), 2010, Page(s): 94 - 95

[51] Goodenough, J.; Aitken, R.，Post-silicon is too late avoiding the $50 million paperweight starts with validated designs，47th ACM/IEEE Design Automation Conference (DAC), 2010, Page(s): 8 - 11

[52] Ahuja, P.K.，Post-silicon validation: It's the unique fails that hurt you， International Test Conference ITC 2009， Page(s): 1

[53] Wenbin Yao, Nianmin Yao, Shaobin Cai, Jun Ni, “Verification Environment for a SCMP Architecture,” In Proceedings of the First International Mutil-Symposiums on Computer and Computational Sciences, pp 787-791, 2006.

[54] 林瞬婷，面向多处理器核 SOC 的软硬件协同验证平台研究，浙江大学，学位论文，2007

[55] Yunji Chen, Weiwu Hu, Tianshi Chen, and Ruiyang Wu. LReplay: A Pending Period Based Deterministic Replay Scheme. In Proc. of ACM/IEEE International Symposium on Computer Architecture (ISCA), June 2010, pp. 187-197.

[56] Yunji Chen, Yi Lv, Weiwu Hu, Tianshi Chen, Haihua Shen, Pengyu Wang, Hong Pan. Fast Complete Memory Consistency Verification. In Proc. of International Symposium on High-Performance Computer Architecture (HPCA), Feb. 2009, pp. 381-392

[57] Borrione, A. Helmy, L. Pierre, and 1. Schmaltz. A Formal Approach to the Verification of Networks on Chip. EURASIP Journal on Embedded Systems, 2009(548324):1-14, February 2009.

[58] Xiaofang Chen, Yu Yang, Ganesh Gopalakrishnan, Ching-Tsun Chou. Efficient methods for formally verifying safety properties of hierarchical cache coherence protocols. Formal Methods in System Design, Volume 36 Issue 1, February 2010

[59] David L. Dill, "A Retrospective on Murphi" in 25 Years of Model Checking (Orna Grumberg & Helmut Veith, eds.), LNCS 5000, pp. 77-88 (2008).

[60] Xiaofang Chen, Yu Yang, Ganesh Gopalakrishnan, Ching-Tsun Chou. Efficient methods for formally verifying safety properties of hierarchical cache coherence protocols. Formal Methods in System Design, Volume 36 Issue 1, February 2010

[61] S.Taylor, C. Ramey, C. Barner, D. Asher, D, “A simulation-based method for the verification of shared memory in multiprocessor systems,” In Proceedings of ICCAD 2001, pp 10-17.

[62] C. Manovit, S. Hangal, “Completely verifying memory consistency of test program executions” In Proceedings of High-Performance Computer Architecture, pp 166-175, 2006.

[63] S. Hangal, D.Vahia, C.Manovit et al, “TSOtool a program for verifying memory systems using the memory consistency model,” In Proceedings of the 31st Annual International Symposium on Computer Architecture, pp 114-123, 2004.

[64] M.M.K. Martin, “Formal verification and its impact on the snooping versus directory protocol debate,” In Proceedings of ICCD, pp 543-549, 2005.

[65] A.A.Sheikh, “Functional verification of multiprocessor based SoC on distributed systems,” In Proceedings of INMIC, pp 33-33, 2002.

[66] S. Matsushita, “Design experience of a chip multiprocessor Merlot and expectation to functional verification,” In Proceedings of the15th International Symposium on System Synthesis, pp 103-108, 2002.

[67] Fu-Ching Yang; Jing-Kun Zhong; Ing-Jer Huang; , "Verifying external interrupts of embedded microprocessor in SoC with on-chip bus," Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on , vol., no., pp.372-377, 10-13 Nov. 2008

[68] van den Broek, T.; Schmaltz, J.; , "Towards a formally verified network-on-chip," Formal Methods in Computer-Aided Design, 2009. FMCAD 2009 , vol., no., pp.184-187, 15-18 Nov. 2009

[69] Mavroidis, I; Papaefstathiou, I; , "Accelerating Emulation and Providing Full Chip Observability and Controllability at Run-Time," Design & Test of Computers, IEEE , vol. PP, no.99

[70] Loitz, S.; Wedler, M.; Brehm, C.; Vogt, T.; Wehn, N.; Kunz, W.; , "Proving Functional Correctness of Weakly Programmable IPs - A Case Study with Formal Property Checking," Application Specific Processors, 2008. SASP 2008. Symposium on , vol., no., pp.48-54, 8-9 June 2008

[71] Hui Wang; Baldawa, S.; Sangireddy, R.; , "Dynamic Error Detection for Dependable Cache Coherency in Multicore Architectures," VLSI Design, 2008. VLSID 2008. 21st International Conference on , vol., no., pp.279-285, 4-8 Jan. 2008

[72] Biswas, S.; Franklin, D.; Sherwood, T.; Chong, F.T.; , "Conflict-Avoidance in Multicore Caching for Data-Similar Executions," Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on , vol., no., pp.80-85, 14-16 Dec. 2009

[73] Ezudheen, P.; Chandran, P.; Chandra, J.; Simon, B.P.; Ravi, D.; , "Parallelizing SystemC Kernel for Fast Hardware Simulation on SMP Machines," Principles of Advanced and Distributed Simulation, 2009. PADS '09. ACM/IEEE/SCS 23rd Workshop on , vol., no., pp.80-87, 22-25 June 2009

[74] Wagner, I.; Bertacco, V.; , "MCjammer: Adaptive Verification for Multi-core Designs," Design, Automation and Test in Europe, 2008. DATE '08 , vol., no., pp.670-675, 10-14 March 2008

[75] Schellhorn, G.; Baumler, S.; , "Formal Verification of Lock-Free Algorithms," Application of Concurrency to System Design, 2009. ACSD '09. Ninth International Conference on , vol., no., pp.13-18, 1-3 July 2009

[76] Ravotto, D.; Sanchez, E.; Reorda, M.S.; Squillero, G.; , "On the generation of test programs for chip multi-thread computer architectures," Test Conference, 2008. ITC 2008. IEEE International , vol., no., pp.1-1, 28-30 Oct. 2008

[77] J.M. Ludden, T.N.Le, W. Roesner et al, “Functional verification of the power4 microprocessor and power4 multiprocessor systems,” IBM J.RES&DEV, vol. 46, NO.1, January 2002.

[78] K. Shimizu, S. Gupta, T. Koyama, et al., “Verification of the Cell Broadband Engine processor,” In Proceedings of the 43rd DAC, pp338-343, 2006

[79] D.W.Victor, J.M.Ludden, R.D.Peterson et al, “Functional verification of the POWER5 microprocessor and POWER5 multiprocessor systems,” IBM J.RES&DEV, vol.49, no.4/5, July/September 2005.

作者简介：

沈海华： 中科院计算所，副研究员 shenhh@ict.ac.cn